

⑩ 日本国特許庁(JP)

⑪ 特許出願公開

⑫ 公開特許公報(A) 平2-48732

⑬ Int. Cl.⁹
G 06 F 9/38

識別記号 庁内整理番号
3 1 0 F 7361-5B

⑭ 公開 平成2年(1990)2月19日

審査請求 有 請求項の数 3 (全7頁)

⑮ 発明の名称 命令パイプライン方式のマイクロプロセッサ

⑯ 特 願 昭63-198789

⑰ 出 願 昭63(1988)8月11日

⑱ 発 明 者 岡 本 光 正 神奈川県川崎市幸区堀川町580番1号 株式会社東芝半導
体システム技術センター内

⑲ 出 願 人 株 式 会 社 東 芝 神奈川県川崎市幸区堀川町72番地

⑳ 代 理 人 弁 理 士 三 好 保 男 外1名

明 細 書

1. 発明の名称

命令パイプライン方式のマイクロプロセッサ

2. 特許請求の範囲

(1) デコードされた命令を実行するため、オペランドアドレス計算部、アドレス変換部、オペランドフェッチ部、演算実行部および汎用レジスタ群を有する命令パイプライン方式のマイクロプロセッサにして、前記演算実行部を重複化して複数の命令を同時に実行できるように構成すると共に、前記汎用レジスタ群をプログラムの流れに従う処理データを格納するレジスタ群と、後続の命令を先回りして実行した結果のデータを格納するレジスタ群とで構成し、逐次的なプログラム処理に従う必要な処理可能な後続命令を先回りして実行可能にしたことを特徴とする命令パイプライン方式のマイクロプロセッサ。

(2) 前記汎用レジスタ群は、さらに後続の命令が先行する命令を飛びこして実行されているかを判定するための比較手段を有していることを

特徴とする請求項1に記載の命令パイプライン方式のマイクロプロセッサ。

(3) 前記複数の各演算実行部は命令レジスタおよび演算手段とを備え、各命令レジスタ内のデータ先行を指定するデスティネーション・フィールドで指定された汎用レジスタ群の特定レジスタに格納されるようにしたことを特徴とする請求項1に記載の命令パイプライン方式のマイクロプロセッサ。

3. 発明の詳細な説明

[発明の目的]

(産業上の利用分野)

本発明は命令並列実行方式のマイクロプロセッサ、特に命令を並列かつ迅速に実行しうる命令パイプライン方式のマイクロプロセッサに関するものである。

(従来の技術)

従来の命令パイプライン方式のマイクロプロセッサにおいては、加算命令などでオペランドのアドレス計算のために特定のレジスタが用いられる

が、その直前の転送命令などにより前記レジスタの内容が変更されるため、汎用レジスタへの到達ステージが終了するまで前記加算命令はオペランドアドレス計算ステージに移れないで命令の処理が遅れる問題があった。

すなわち、第7図は従来の命令パイプライン方式のマイクロプロセッサの概略構成を示す。

同図において、1はマイクロプロセッサPと外部回路を接続するためのバス制御部(BCU)、2は命令フェッチ部(IFU)、3は命令を解釈するためのデコーダ(DEC)、4はオペランドアドレスを計算するためのオペランドアドレス計算部(OAG)、5は論理アドレスを物理アドレスに変換するためのアドレス変換部(AT)、6はオペランドをフェッチするためのオペランドフェッチ部(OPF)、7は命令を実行する演算実行部(EXU)、8は複数のレジスタR₁、R₂、R₃、R₄、…(図示せず)からなる汎用レジスタ群(GR)を示す。

第7図に示すマイクロプロセッサPにより、例

が完全に実行されてからでないで後続の命令は実行されえなかった。

したがって、従来の命令パイプライン方式のマイクロプロセッサにおいては処理の流れが停滞しパイプライン方式の利点を生かしきれていなかった。

そこで、本発明は、上記に露みてなされたものであり、その目的とするところは、先行する命令の実行結果を持たずに後続の命令を先回りして実行することができる命令パイプライン方式のマイクロプロセッサを提供することである。

[発明の構成]

(課題を解決するための手段)

上記目的を達成するために、本発明に従う命令パイプライン方式マイクロプロセッサにおいては、演算実行部をメモリオペランドをもたない命令を実行する第1の演算実行部と、メモリオペランドをもつ命令を実行する第2の演算実行部と、浮動小数点命令を実行する第3の演算実行部とで構成すると共に、汎用レジスタを、プログラムの

例えば第6図に示すプログラム命令を実行した場合の命令パイプライン処理のタイミングを第8図に示す。第6図および第8図に示す如く、アドレスAの内容を汎用レジスタ群8内の図示しないレジスタR₁へ転送する命令a₁とし、汎用レジスタ群8内のR₃レジスタのデータをR₂へ転送する命令をa₂、上記レジスタR₂で修飾される如くアドレスBの内容をレジスタR₄へ転送する加算命令をa₃、レジスタR₂のデータをアドレスCで示されるメモリへ転送する命令をa₄とすると、命令a₃でオペランドのアドレス計算を行なうためにレジスタR₂の内容が用いられるが、命令a₂でレジスタR₂が変更されてしまう。したがって命令a₂の汎用レジスタ群8への転送が(雷込み)が終了するまで命令a₃はオペランドアドレス計算部へ入れず該命令の処理が遅れてしまう。

(発明が解決しようとする課題)

すなわち、従来のマイクロプロセッサにおいては演算実行部および汎用レジスタが重複化されていないために汎用レジスタ群8を更新する命令

流れに沿った処理データを記憶するレジスタ群と、処理可能な命令を先取り実行した結果を記憶するレジスタ群とで構成している。

(作用)

演算実行部を重複化すると共に汎用レジスタ群の構成を、プログラムの流れに沿った処理データを記憶するレジスタ群と、処理可能な命令を先取り実行した結果を記憶するレジスタ群とに重複化することにより、汎用レジスタ群のデータ更新を行ない、先行する命令の実行結果を持たずに後続の命令を先回りして実行し、マイクロプロセッサの性能を上げるようにしている。

(実施例)

第1図は本発明による命令パイプライン方式のマイクロプロセッサの原理構成図を示す。同図において、第7図のものと同一の構成要素は同じ参照番号で示してある。第1図の命令パイプライン方式マイクロプロセッサにおいては、演算実行部をメモリオペランドをもたない命令を実行する第1の演算実行部10(SEP)、メモリオペラ

ンドをもつ命令を実行する第2の演算実行部11(IEP)、浮動小数点命令を実行する第3の演算実行部12(FEP)で構成している。なお、13は解決(デコード)された命令を前記各演算実行部へ送出する命令送出部、14は浮動小数点レジスタを示す。

第1図に示す本発明によるマイクロプロセッサにおいては第6図に示す同じパイプライン命令に対して第2図に示すように実行する。

すなわち、毎クロックごとに命令がフェッチされ処理されてゆくが、命令 a_1 はメモリオペランドをもつので命令フェッチ部2→デコーダ3→命令送出部13→オペランドアドレス部4→アドレス変換部5→オペランドフェッチ部6→第2の演算実行部11と処理される。命令 a_2 はメモリオペランドをもたない(レジスタオペランドのみ)命令であるので、命令フェッチ部2→デコーダ3→命令送出部13→第1の演算実行部10と処理される。第2図に示すように命令 a_2 と a_1 の実行が第5クロックおよび第6クロックで終了して

しまうので汎用レジスタ群8'内のデータとプログラムの流れに不一致が生じるのを避けるために汎用レジスタ群8'は後述するようにプログラムの処理結果データを保持するレジスタ群と、命令の実行結果を一時的に先回りして保持するレジスタ群とに重複化して解決している。

次に、第3図を参照して、第1図の汎用レジスタ8'および第1、第2、第3の演算実行部10、11、12の詳細な構成を示す。

第3図において、第1の演算実行部10は第1の演算器20と第1の命令レジスタ21を、第2の演算実行部11は第2の演算器22と第2の命令レジスタ23、および第3の演算実行部12は第3の演算器24と第3の命令レジスタ25を有している。そして前記各命令レジスタ21、23、25のフォーマットは第4図のようになっている。

すなわち、第4図においてOPは関連する演算器の演算動作指定フィールド、SRはソースレジスタ指定フィールド、DRはデスティネーションレジスタ指定フィールド、SA/IDはソースオ

ペランドアドレスまたはイミディエイトデータ保持フィールド、DAはデスティネーションオペランドアドレス保持フィールド、PCは各演算器で実行中の命令アドレス保持フィールドを示す。

第3図に戻って、本発明による汎用レジスタ群8'はプログラムに従って処理された結果のデータを保持するレジスタ群CGRi($i=1, 2, 3 \dots n$)と命令が実行された演算器からの演算結果を格納するレジスタ群FGRi($i=1, 2, 3 \dots n$)に分けられている。なお、第3図で30は命令アドレス比較部を示し、これは汎用レジスタ群8'に含まれていると見なしてもよい。

第5図はレジスタ群FGRのフォーマットの詳細を示す、各3ビットからなるタグ部F、I、Sと各データを保持するデータ保持部FGRiからなっている。

第3図に戻って、各命令レジスタ21、23、25のPCフィールドは、後続の命令が先行する命令を飛越して実行されているかどうかを判定するために前記命令アドレス比較部30へ入力され、

前記各PCフィールドを比較している。比較結果から第1の命令レジスタ21のPCフィールドが最小でないと出力 C_1 が"1"に、第2の命令レジスタ23のPCが最小でないと出力 C_2 が"1"、そして第3の命令レジスタ25のPCが最小でないと出力 C_3 が"1"となるような出力が前記比較部30から出される。

更に前記アドレス比較部30の出力 C_4 (FGR囲込み信号)は命令を実行した演算部の命令レジスタのアドレスPCが最小でない場合に"1"となり、最小の場合に前記比較部30の出力 C_5 (CGR囲込み信号)は"1"となるように構成されている。

また各演算部の演算終了時に、レジスタFGR中のタグSは $C_4=1$ 、 $C_1=1$ のとき"1"にセットされ、Iタグは $C_2=1$ のとき"1"に、そしてFタグは $C_3=1$ のときに"1"にセットされるようになっている。

各演算器20、22、24の演算結果は、各命令レジスタ21、23、25のDRフィールドで

指定されたデータ保持部FGR1に格納されるようになっている。

各演算器20, 22, 24の演算終了時に、 $C_4 = 0$ 、 $C_5 = 1$ の場合、各演算結果は各命令レジスタのDRフィールドで指定されるデータ保持部FGR1およびCGR1に格納されると共に、F, I, Sタグのいずれかがセットされている別のFGR1のデータは対応するCGR1に指定されるようになっている。

したがって第2図に示すように、動作時ににおいて、命令 a_2 が実行される第5クロックで、第3図に示す第1の命令レジスタ21のPCフィールドが命令 a_2 のアドレスを有し、第2の命令レジスタ23のPCフィールドは命令 a_1 のアドレスを有しているが、第3の命令レジスタ25のPCフィールドには命令送出部13から命令が送出されてないのでそのPCフィールドにはアドレスがない。したがって第3の演算実行部12では何らの命令の実行も行なわれない。

一方、前記のように第1の命令レジスタ21に

は a_2 が入っているのでそのアドレスPCフィールドは、第2の命令レジスタ23に入っている命令 a_1 のアドレスPCフィールドよりも大であるから $C_1 = 1$ となり演算結果がFGR2へ回送されると共にタグSが1にセットされる。

第6クロックにおいては、第2の演算部11で命令 a_1 が実行されるが、このとき、第1の演算部10内の命令レジスタ21では上記のように命令 a_2 がすでに実行され終っていて命令がないので何らの動きもしない。したがって $C_4 = 0$ 、 $C_5 = 1$ となるのでCGR1およびFGR1に命令 a_1 の演算結果が格納されると共にタグS₁が1となっているFGR2のデータがCGR2へ転送され、プログラムの流れに沿った処理データとして用いられるように記憶される。

上記の動作から判るように、次の命令 a_3 の実効アドレスの算出に必要なデータは、第5クロックの終了時にFGR2(R₂レジスタに対応する)から取出すことができるので命令 a_3 は、直ちに実効アドレス計算部4へ進むことができる。した

がって、従来のこの種のマイクロプロセッサのように a_1 、 a_2 の実行終了を待ってから命令 a_3 の処理に入るというような逐次的なプログラム処理に従う必要がない。

〔発明の効果〕

以上述べたように本発明においては、演算実行部を重複化して複数の命令を同時に実行できるように構成すると共に、汎用レジスタ群もプログラムの流れに従う処理データを格納するレジスタ群と、後続の命令を先回りして実行した結果のデータを格納するレジスタ群とを有するように重複化構成としている。

したがって本発明における命令パイプライン方式のマイクロプロセッサにおいては、先行する命令の実行結果を待たずに後続の命令を先回りして実行できるのでマイクロプロセッサの性能を著しく向上することができる。

4. 図面の簡単な説明

第1図は、本発明による命令パイプライン方式のマイクロプロセッサの基本構成図、

第2図は、第1図のマイクロプロセッサを構成する各部の命令処理流れ図、

第3図は、第1図の演算実行部および汎用レジスタ群の詳細な構成図、

第4図は、第3図の各命令レジスタのフォーマット図、

第5図は、汎用レジスタ群内の命令を先回り処理した結果を格納するレジスタ群FGRの入力フォーマット、

第6図は、従来のマイクロプロセッサの有する問題点を明らかにするプログラム例、

第7図は、従来技術による命令パイプライン方式のマイクロプロセッサの構成図、

第8図は、第7図のプロセッサ内の命令処理の流れ図、をそれぞれ示す。

1…バス制御部、

2…命令フェッチ部、

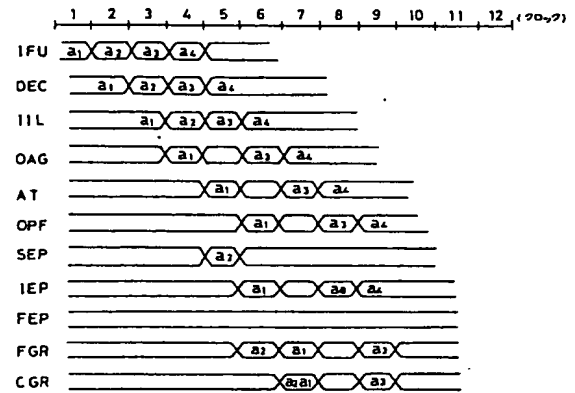
3…デコーダ、

4…オペランドアドレス計算部、

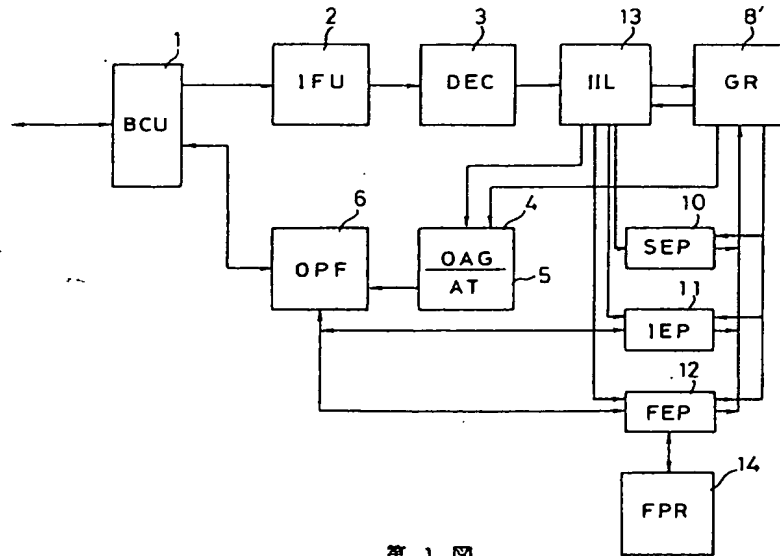
5…論理アドレス変換部、

- 6 … オペランドフェッチ部、
- 8 … 汎用レジスタ群、
- 10 … 第1の演算実行部、
- 11 … 第2の演算実行部、
- 12 … 第3の演算実行部、
- 13 … 命令送出部、
- 14 … 浮動小数点レジスタ。

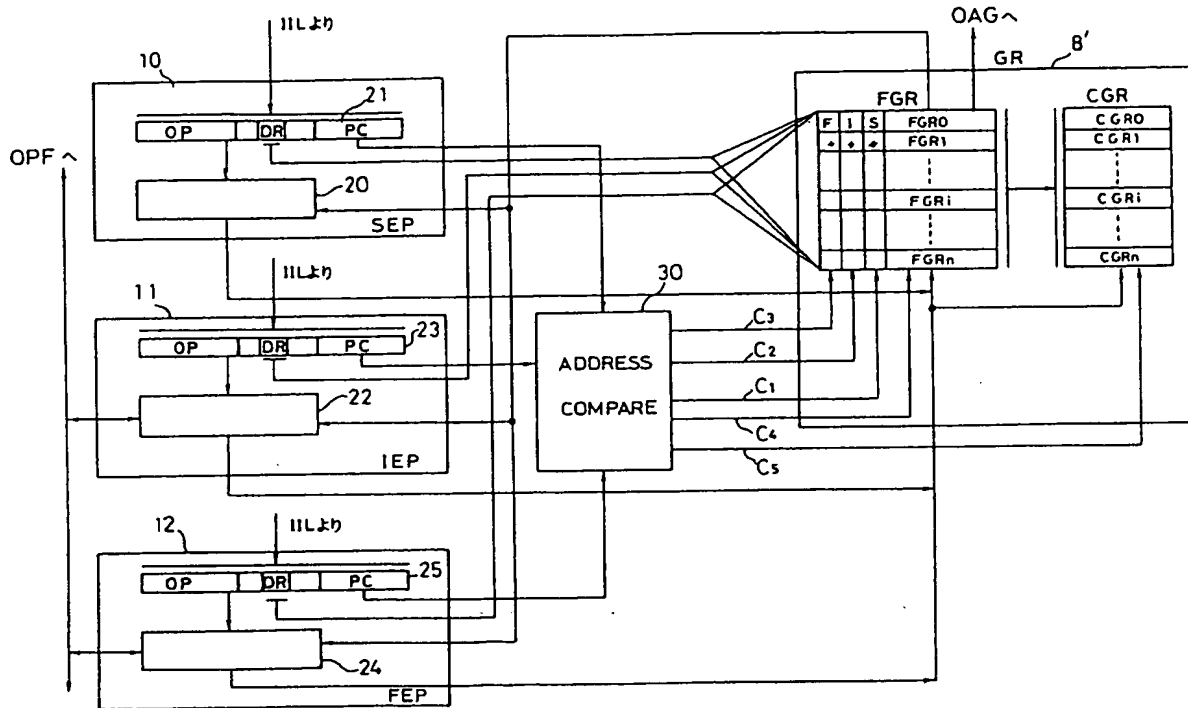
代理人弁護士 三好保男



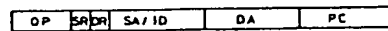
第2図



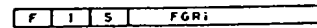
第1図



第 3 図



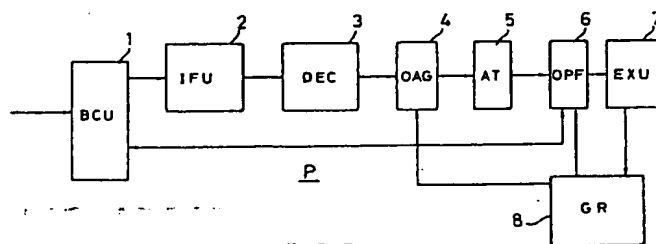
第 4 図



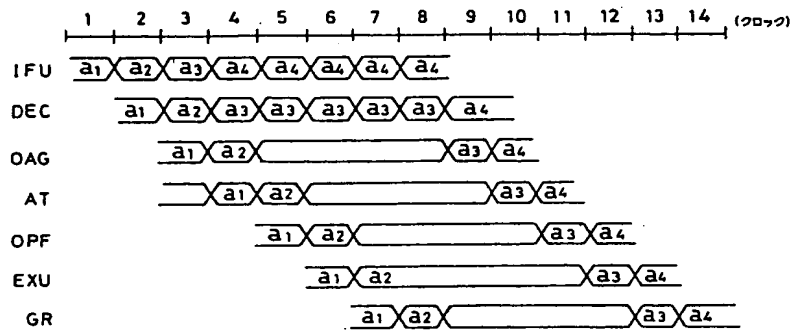
第 5 図

a_1 -----MOV $addr\ A, R_1$: $addr\ A \leftarrow R_1$
 a_2 -----MOV R_3, R_2 : $R_3 \leftarrow R_2$
 a_3 -----ADD $addr\ B(R_2), R_4$: $addr\ B(R_2) + R_4 \leftarrow R_4$
 a_4 -----MOV $R_2, addr\ C$: $R_2 \leftarrow addr\ C$

第 6 図



第 7 図



第 8 図

(19) Japan Patent Office (JPO)

(11) Publication Number

(12) Publication of Unexamined Patent Application (A) H2-48732

(51) International Classification⁵ G 06 F 9/38

Identification Number: 310 F

JPO file number: 7361-5B

(43) Date of Publication of Application: 19 February 1990

Request for examination Not requested Number of Claims: 3 (total 7 pages)

(54) Title of the invention

Micro Processor of Instruction Pipeline System

(21) Application filing No. 63-198789

(22) Application filing date August 11, 1988

(72) Inventor(s) OKAMOTO MITSUMASA; TOSHIBA CORPORATION,
SEMICONDUCTOR SYSTEM ENGINEERING DIVISION; 580-1, Horikawa-cho,
Saiwai-ku, Kawasaki, Kanagawa, Japan

(71) Applicant(s) TOSHIBA CORPORATION; 72, Horikawa-cho, Saiwai-ku,
Kawasaki, Kanagawa, Japan

(74) Agency MIYOSHI YASUO, Patent Attorney and & 1 other

Specification

1. Title of the Invention

MICRO PROCESSOR OF INSTRUCTION PIPELINE SYSTEM

2. What is Claimed is:

[Claim 1] A micro processor of instruction pipeline system, comprising: an operand address generator; an address transducer; an operand fetch unit; and an operation execution unit and a general-purpose register group in order to execute decoded instructions, wherein said operation execution unit is configured to be duplicated and simultaneously execute a plurality of instructions; and said general-purpose group is composed of a register group for storing processed data in accordance with the flow of a program and a register group for storing data of a result obtained by executing a subsequent instruction in advance, thereby enabling the processable subsequent instruction to be executed in advance without following sequential program processing.

[Claim 2] The micro processor of instruction pipeline system according to Claim

1, wherein said general-purpose register group further comprises a comparing means for judging whether or not the subsequent instruction is executed in advance of a preceding instruction.

[Claim 3] The micro processor of instruction pipeline system according to Claim 1, wherein each of a plurality of said operation execution unit comprises an instruction register and an operation means so as to perform storage in a specific register of the general-purpose register group specified in a destination field of the instruction register that specifies data precedence.

3. Detailed Description of the Invention

[Object of the Invention]

(Field of Industrial Application)

The present invention relates to a micro processor of instruction parallel execution system and more particularly to a micro processor of instruction pipeline system which can execute instructions in parallel and fast.

(Description of the Related Art)

In a conventional micro processor of instruction pipeline system, a specific register has been used for operand address calculation in an addition instruction or the like, but there has been a problem that since the contents of said register are modified by the last transfer instruction or the like, said addition instruction cannot transit to an operand address calculation stage until a write stage to a general-purpose register is finished, causing the processing of the instruction to delay.

Specifically, Fig. 7 shows a schematic configuration of a conventional micro processor of instruction pipeline system.

In this figure, a reference numeral 1 denotes a bus control unit (BCU) for connecting a micro processor P and an external circuit, a reference numeral 2 denotes an instruction fetch unit (IFU), a reference numeral 3 denotes a decoder (DEC) for decoding an instruction, a reference numeral 4 denotes an operand address generator (OAG) for calculating an operand address, a reference number 5 denotes an address transducer (AT) for transducing a logical address to a physical address, a reference numeral 6 denotes an operand fetch unit (OPF) for fetching an operand, a reference numeral 7 denotes an operation execution unit (EXU) for executing an instruction, and a reference numeral 8 denotes a general-purpose register group (GR) consisting of a plurality of registers $R_1, R_2, R_3, R_4, \dots$ (not shown).

Fig. 8 shows the timing of instruction pipeline processing in the case where, for example, program instructions shown in Fig. 6 are executed by the micro processor P shown in Fig. 7. As shown in Figs. 6 and 8, suppose an instruction to transfer contents of an address A to the register R_1 in the general-purpose register group 8 (not shown) is indicated by a reference sign a_1 , an instruction to transfer data of the register R_3 to R_2 in the general-purpose register group 8 is indicated by a reference sign a_2 , an addition instruction to transfer contents of an address B modified in the above-mentioned register R_2 to the register R_4 is indicated by a reference sign a_3 , and an instruction to transfer data of the register R_2 to a memory indicated by an address C is indicated by a reference sign a_4 . At this time, although the contents of the register R_2 are used to

calculate the operand address by the instruction a₃, the register R₂ is changed by the instruction a₂. Accordingly, the instruction a₃ cannot enter the operand address generator until the transfer of the instruction a₂ to the general-purpose register group 8 (write) is finished, which delays the processing of the instruction a₃.

(Problems to be Solved by the Invention)

In other words, in the conventional micro processor, since the operation execution unit and the general-purpose registers are not duplicated, a subsequent instruction cannot be executed before an instruction which updates the general-purpose register group 8 is completely executed.

Accordingly, in the conventional micro processor of instruction pipeline system, its processing flow stagnates and advantages of the pipeline system have not been exerted sufficiently.

The present invention seeks to solve the above-mentioned problem. The object is to provide a micro processor of instruction pipeline system capable of executing a subsequent instruction in advance without waiting for an execution result of a preceding instruction.

[Constitution of the Invention]

(Means for Solving the Problem)

In order to achieve the above-mentioned object, in a micro processor of instruction pipeline system according to the present invention, an operation execution unit is composed of a first operation execution unit for executing an instruction with no memory operand, a second operation execution unit for executing an instruction with a memory operand, and a third operation execution unit for executing a floating-point instruction, and general-purpose registers are composed of a register group for storing processed data in accordance with the flow of a program and a register group for storing a result obtained by executing a processable instruction in advance.

(Action)

The operation execution unit is duplicated and the configuration of the general-purpose register group is also duplicated by consisting of the register group for storing the processed data in accordance with the flow of the program and the register group for storing the result obtained by executing the processable instruction in advance to thereby update the data in the general-purpose register group and to executing the subsequent instruction in advance without waiting for the execution

result of the preceding instruction, which improves performance of the micro processor.

(Preferred Embodiment of the Invention)

Fig. 1 is a fundamental block diagram showing a micro processor of instruction pipeline system according to the present invention. In this figure, like components are indicated by the same reference numerals and signs as those in Fig. 7. In the micro processor of instruction pipeline system in Fig. 1, the operation execution unit is composed of a first operation execution unit 10 (SEP) for executing an instruction with no memory operand, a second operation execution unit 11 (IEP) for executing an instruction with a memory operand, and a third operation execution unit 12 (FEP) for executing a floating-point instruction. Incidentally, a reference numeral 13 denotes an instruction delivery-unit for delivering a decoded instruction to each-of said operation execution units, and a reference numeral 14 denotes a floating-point register.

In the micro processor according to the present invention as shown in Fig. 1, the same pipeline instructions as those of Fig. 6 are executed as shown in Fig. 2.

Specifically, an instruction is fetched at every clock to be processed, and the instruction a₁, having a memory operand, is processed from the instruction fetch unit 2, via the decoder 3, the instruction delivery unit 13, the operand address generator 4, the address transducer 5, and the operand fetch unit 6, to the second operation executing unit 11. The instruction a₂, having no memory operand (having only a register operand), is processed from the instruction fetch unit 2, via the decoder 3, and the instruction delivery unit 13, to the first operation execution unit 10. As shown in Fig. 2, the executions of the instructions a₂ and a₁ are finished in the fifth clock and the sixth clock and, therefore, in order to prevent discrepancy between data within a general-purpose register group 8' and the flow of a program, the general-purpose register group 8' is duplicated by consisting of a register group holding processing result data of the program and a register group temporarily holding an execution result by executing an instruction in advance as shown described later.

Next, referring to Fig. 3, the configurations of the general-purpose register 8' and the first, second, third operation execution units 10, 11 and 12 in Fig. 1 are described in detail.

In Fig. 3, the first operation execution unit 10 has a first operator 20 and a first instruction register 21, the second operation execution unit 11 has a second operator 22 and a second instruction register 23, and the third operation execution unit 12 has a third operator 24 and a third instruction register 25. The format of each of said instruction registers 21, 23 and 25 is as shown in Fig. 4.

Specifically, in Fig. 4, OP denotes an operation behavior specifying field of the relevant operator, SR denotes a source register specifying field, DR denotes a destination register specifying field, SA/ID denotes a source operand address or immediate data holding field, DA denotes a destination operand address holding field, and PC denotes an instruction address holding field for an instruction being executed in each operator.

Back to Fig. 3, the general-purpose register group 8' according to the present invention is divided into a register group CGR_i ($i = 1, 2, 3 \dots n$) holding data of a result processed in accordance with a program and a register group FGR_i ($i = 1, 2, 3 \dots n$) storing an operation result from an operator executing an instruction. Incidentally, in Fig. 3, a reference numeral 30 denotes an instruction address comparing unit, which may be considered to be included in the general-purpose register group 8'.

Fig. 5 shows the format of the register group FGR in detail. The register group FGR consists of a tag units F, I, and S each consisting of 3 bits, and a data holding unit FGR_i for holding each data.

Back to Fig. 3, the PC fields of the respective instruction registers 21, 23 and 25 are inputted to said instruction address comparing unit 30 and said respective PC fields are compared to judge whether a subsequent instruction is executed in advance of a preceding instruction. Said comparing unit 30 outputs in such a manner based on the comparison result, that if the PC field of the first instruction register 21 is not the least, an output C₁ becomes "1", if the PC field of the second instruction register 23 is not the least, an output C₂ becomes "1", and if the PC field of the third instruction register 25 is not the least, an output C₃ becomes "1".

Furthermore, an output C₄ (an FGR write signal) of said address comparing unit 30 becomes "1" in the case where the address PC of the instruction register of an operator executing an instruction is not the least, while an output C₅ (a CGR write signal) of said comparing unit 30 becomes "1" in the case where said address PC is the least.

In addition, when the operation in each operator is finished, the tag S in the register FGR is designed to be set to "1" in the case of C₄ = 1 and C₁ = 1, the tag I to "1" in the case of C₂ = 1, and the tag F to "1" in the case of C₃ = 1.

The operation results of the respective operators 20, 22 and 24 are designed to be stored in the data holding units FGR_i specified in the DR fields of the respective instruction registers 21, 23 and 25.

When the operations in the respective operators 20, 22 and 24 are finished, in the case of C₄ = 0 and C₅ = 1, operation results are designed to be stored in the data

holding units FGR_i and CGR_i specified in the DR fields of the instruction registers, respectively, and the other data of FGR_i with any of the tags F, I and S set is designed to be specified to the corresponding CGR_i.

Accordingly, as shown in Fig. 2, during operation, at the fifth clock when the instruction a₂ is executed, the PC field of the first instruction register 21 shown in Fig. 3 has the address of the instruction a₂, the PC field of the second instruction register 23 has the address of the instruction a₁, and the PC field of the third instruction register 25 has no address since the instruction delivery unit 13 has not delivered any instruction. Accordingly, the third operation execution unit 12 does not execute any instruction.

On the other hand, as mentioned above, since the first instruction register 21 contains a₂, its address PC field is more larger than the address PC field of the instruction a₁ contained in the second instruction register 23, C₁ becomes 1, so that the operation result is written to FGR₂ and the tag S is set to 1.

At the sixth clock, the instruction a₁ is executed in the second operation execution unit 11. At this time, the instruction register 21 in the first operation execution unit 10 has no instruction as the execution of the instruction a₂ has been finished and therefore, takes no action. Accordingly, C₄ is set to 0 and C₅ is set to 1, and the operation result of the instruction a₁ is stored in CGR₁ and FGR₁ and the data of FGR₂ with the tag S₁ set to 1 is transferred to and stored in CGR₂ so as to be used as processed data in accordance with the flow of the program.

As understood in the above-mentioned operations, since the data necessary for calculating the effective address of the next instruction a₃ can be derived from the FGR₂ (corresponding to the R₂ register) at the end of the fifth clock, the instruction a₃ can immediately proceed to the effective address generator 4. Accordingly, it is not necessary to follow the sequential program processing as in this type of conventional micro processor in which the instruction a₃ starts to be processed after the executions of a₁ and a₂ are finished.

[Effects of the Invention]

As described above, according to the present invention, the operation execution unit is configured to be duplicated and to simultaneously execute a plurality of instructions, and the general-purpose register group is also configured to be duplicated by consisting of the register group for storing processed data in accordance with the flow of a program and the register group for storing data of a result by executing a subsequent instruction in advance.

Accordingly, in the micro processor of instruction pipeline system according to

the present invention, since the subsequent instruction can be executed in advance without waiting for the execution result of the preceding instruction, the performance of the micro processor can be remarkably improved.

4. Brief Description of the Drawings

[Fig. 1]

Fig. 1 is a fundamental block diagram showing a micro processor of instruction pipeline system according to the present invention.

[Fig. 2]

Fig. 2 is a flow diagram showing instruction processing of respective units composing the micro processor of Fig. 1.

[Fig. 3]

Fig. 3 is a detailed block diagram showing an operation execution unit and a general-purpose register group of Fig. 1.

[Fig. 4]

Fig. 4 is a format diagram of each instruction register of Fig. 3.

[Fig. 5]

Fig. 5 shows an input format of a register group FGR for storing a result obtained by processing an instruction in advance in the general-purpose register group.

[Fig. 6]

Fig. 6 shows a program example clarifying a problem that a conventional micro processor has.

[Fig. 7]

Fig. 7 is a block diagram showing a conventional micro processor of instruction pipeline system according to the related art.

[Fig. 8]

Fig. 8 is a flow diagram showing instruction processing in the processor of Fig. 7.

1 bus control unit

2 instruction fetch unit

3 decoder

4 operand address generator

5 logical address transducer

6 operand fetch unit

8' general-purpose register group

- 10 first operation execution unit
- 11 second operation execution unit
- 12 third operation execution unit
- 13 instruction delivery unit
- 14 floating-point register

Patent agency, MIYOSHI YASUO